REMARKS

Claims 1-14, 17, 20-50, 62-64 are pending and stand rejected. In response, claims 1-14, 17, 2-48, and 62-63 are amended and claims 65-67 are added. Claims 1-14, 17, 20-50, and 62-67 are pending upon entry of this amendment. Support for the new claims is found throughout the specification, including at paragraphs 24 and 98-99.

Claim Objections

Claims 2-14, 17, 20-35, 37-48, and 62 are objected to because of informalities. Specifically, the dependent claims are objected to because the claims refer to the independent claims by stating "A method according to claim..." and "A computer system according to claim..." instead of "*the* method" and "*the* computer system." While Applicants do not agree that amendment is necessary, Applicants have amended the claims in accordance with the Examiner's request in order to expedite prosecution.

35 U.S.C. § 102(b) Rejections

Claims 1-9, 20-26, 28-50 and 62-64 stand rejected under 35 U.S.C. § 102(b) as being anticipated by Thilmany et al. ("BizTalk®: Implement Design Patterns for Business Rules with Orchestration Designer"). Applicants respectfully traverse this rejection as applied to the amended claims.

Amended claim 1 recites a method of creating an application for executing on at least one machine having a memory. The method comprises creating a definition of a node and a specification defining how the node can interact with other nodes,

> wherein the processing of the machine readable data file, in the run time environment, **dynamically interconnects** each node according to the specification and/or a

> data input such that data input to the application is processed by the at least one node and, if further processing is required, forwarded to other nodes for that further processing.

Thus, a node is dynamically interconnected when processed in a run time environment according the definition provided in the specification and/or according to a definition provided by a data input. The dynamic interconnection capability allows nodes to be interconnected and reconnected at run time based on inputs to provide new or additional capabilities. Support for these amendments are found throughout the specification and specifically at paragraphs 0016, and 0096-0103.

Thilmany does not disclose that processing of a machine readable data file, in a run time environment, dynamically interconnects each node according to the specification and/or by a data input as claimed. Thilmany describes an orchestration designer tool that can be used to organize a set of BizTalk objects. The orchestration designer is based on Microsoft Visio® and allows a developer to use high-level techniques to orchestrate (organize) the objects. The developer uses flowchart shapes to define business logic among the objects such as conditional looping and branching execution. *See* page 5, text under headings "BizTalk and the Orchestration Designer" and "Using the BizTalk Orchestration Designer."

However, Thilmany neither teaches nor suggests that the business logic connecting the objects is defined *dynamically* in a run time environment Rather, a developer uses Visio® to establish connections between objects that remain fixed at run time. Accordingly, Thilmany does not disclose dynamically interconnecting each node according to the definition provided in the specification and/or a definition provided by a data input, as claimed.

In addition, Thilmany does not teach creating a definition of at least one node and specification, and holding the definition and specification in at least one machine readable data file, where **that** machine readable data file is processed in the run time environment. Thilmany

teaches that the BizTalk Orchestration Designer is used to produce a document that must then be compiled in order to generate the XML XLANG specification. *See* page 5, text under headings "BizTalk and the Orchestration Designer."

Moreover, Thilmany does not teach a specification defining "resources useable by the at least one node during processing of the machine readable data file." The portion of Thilmany relied upon by the Examiner to reject this element reads:

> There are three key object categories in the business process page of the Orchestration Designer: flowchart shapes, ports, and implementation shapes. Conceptually, the way this works is that you, or your friendly neighborhood business analyst, use the flowchart shapes to create the process flow. The flowchart shapes support basic constructs such as if...then...else decisions, looping while a condition is true, branching execution, rejoining the branches together, and defining transactional boundaries.

This does not teach resources useable during processing of the at least one machine readable data file. It teaches resources usable during design of the system.

Applicants respectfully submit that the claimed invention is not anticipated by the cited reference for the reasons described above. The other independent claims are not anticipated for at least the same reasons. Accordingly, Applicants request that the Examiner withdraw the § 102 rejection and allow the claims.

Claim 4 depends from claim 1 and recites that the method further comprises arranging a node "to comprise a plurality of layers, each layer being arranged to perform a predetermined function." The node is defined by the machine readable data file and is processed in the run time environment as stated in claim 1.

The Examiner bases the rejection of this claim on the section on page 5 of Thilmany under the heading "BizTalk and the Orchestration Designer" where it states:

> "the designer provides a high level means of orchestrating a middleware system's moving parts; those moving parts can take the form of a message queue, COM component, Biztalk Channel, file, e-mail message, or HTTP based service;

> Orchestration Designer provides a way to create loosely coupled business processes that may optionally be long-running in nature."

This portion does not disclose arranging a node to comprise a plurality of layers as claimed. Rather, it simply discloses that BizTalk may use a plurality of different transport mechanisms. Thilmany states that these mechanisms may be changed at design time and then compiled to a XLANG schedule instance. But the reference does not use the same file for design and run time definitions. Thus, Thilmany does not disclose arranging the layers of a node as defined by a machine readable data file that is processed in a run time environment as claimed.

Claim 5 depends from claim 4 and further recites arranging the layers of the nodes to be interchangeable. This rejection is based on the same portion of Thilmany used to reject claim 4. The cited portion does not disclose that the "moving parts" are interchangeable. For instance, to change from one transport mechanism to another could entail modification of other aspects of BizTalk, or indeed wholesale restructuring. The level of change required to move from one transport mechanism to another is simply not discussed in the reference. Accordingly, Applicants respectfully submit that claims 4 and 5 are not anticipated by Thilmany.

Applicants submit that the claims are patentable for the reasons described above. The claims not specifically mentioned above incorporate the elements of their respective base claims and are patentable for at least the same reasons. Accordingly, Applicants request that the Examiner allow the application and pass it to issue. The Examiner is invited to contact the undersigned to advance the prosecution of this case.

Respectfully submitted,
ABDUL KAYAM ET AL.

Dated: January 3, 2008

By: /Brian Hoffman/
Brian M. Hoffman, Reg. No. 39,713
Attorney for Applicant
Fenwick & West LLP
801 California Street
Mountain View, CA 94041
Tel.: (415) 875-2484
Fax: (415) 281-1350